Making of Advanced Ontology based Search and Standardization Engine using Type Convertor Algorithm

Rajalakshmi. A[1], Priya Radhika Devi. T[2]

[1, 2]Mailam Engineering College, Department of Computer Science and Engineering, Tamil Nadu, India

Abstract: ***Semantic web service (SWS) is an extension of the web service with an explicit representation of meanings. It promises to increase the level of automation and has ability to integrate and reuse diverse information resources relevant to a given situation in a cost-effective way. It has the potentiality to change the way of knowledge and business services which are consumed and provided on the web. In conventional web service the problem of discovering and selecting the most suitable web service represents a challenge for SWSs. We propose Type Convertor algorithm that converts WSDL files into OWL-S files and facilitates the redefinition of the conventional web service annotations (WSDL) using semantic annotations (OWL-S). The aim of this study is to achieve a standardization search engine based on advanced ontology languages and using WSDL to OWL-S type convertor. An ontology enhanced search engine or an ontology enhanced service registry is able to do the search automatically and in most efficient way.***

Keywords: Semantic Web Service, Ontology, WSDL, OWL-S, Standardization Engine

1. Introduction

The Semantic Web Service [1] is an extension of the current web in which information is given well- defined meaning, better enabling computers and people to work in co-operation. Since the existing technologies for web services only describe in syntactical level, it is difficult for service requestors and service providers to interpret or represent non- trivial statements such as the meaning of inputs and outputs or applicable constraints. Semantic description of web services can make possible for automatic discovery, composition and execution across heterogeneous users and domains. Ontologies are the key means to achieve this functionality. They are used to annotate unstructured information with semantic information, to integrate information and to generate user specific views that make knowledge access easier The Web Ontology Language (OWL) is a new formal language for representing ontologies in the Semantic Web [2]. OWL has features from several families of representation languages, including primarily Description Logics and frames. OWL also shares many characteristics with RDF, the W3C base of the Semantic Web.

In [3], We introduce our new discovery mechanism .This mechanism has the ability to provide optimal results for any service request. This mechanism is distinguished by its service repository, which is built using the advertisements of semantic and non SWS. In addition its advertisements improve the speed and quality of the discovery process.Fig.1 shows the discovery mechanism system organization .we can summaries our mechanisms into two phases:

Phase 1 “Database Creation”: in this phase the database of the discovery mechanism is created using the semantic definition of the registered services, it consists of three main steps:

1. Mapping from the WSDL [4] of the already existing WSs to a semantic definition using OWL-S [5], [6].
2. Registering all the available semantic definitions (whether they belong to WSs or SWSs) in the “Unclassified Profiles” database.
3. Classifying these data into prepared clusters to make the discovery easier and faster.

Phase 2 “Discovery Process”: this phase starts with receiving the user request of a certain service and then follows these steps:

1. Use the discovery algorithm to search into the database for the suitable results.
2. Use the ranking algorithm to rank the results to enhance the user selections.

This study aims to redefine the conventional web services using semantic markups. This does not only mean the process of converting the conventional web service description language (WSDL) to a semantic one (i.e., OWL-S), but it also means the standardization of this definition by using the concept of ontology to describe any type of data in the service. Consequently, the proposed algorithm contains an important component called the ontology search and standardization engine (OSSE) that helps in the standardization process. OSSE’s function is based on searching for a suitable ontology in the “local ontology repository.”

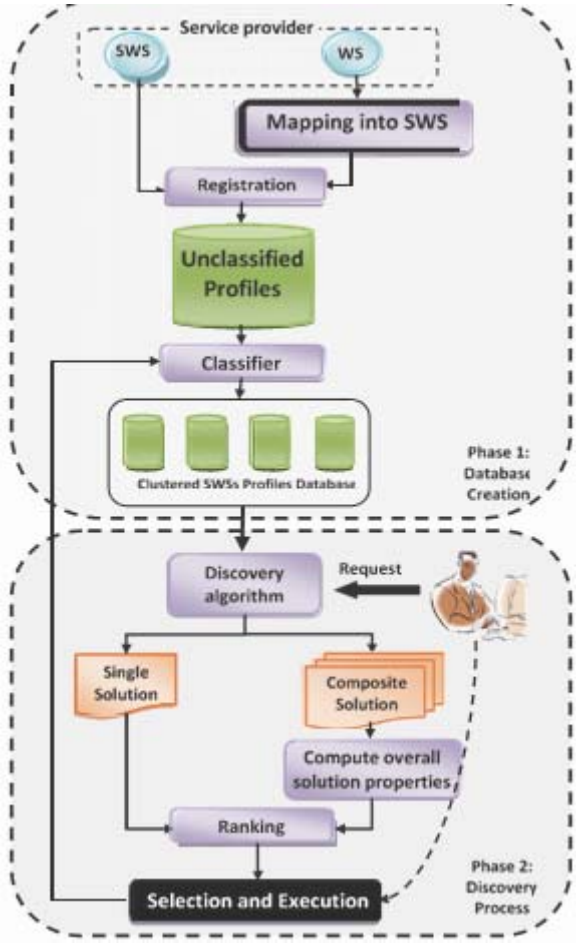

Figure 1: Discovery Mechanism System Organization

2. Web Service Definition Language

WSDL [7] is an XML [8] format used to describe network services as a set of endpoints operating on messages containing either document-oriented or procedure-oriented information. WSDL is often used in combination with SOAP [9] and an XML Schema [10] to provide web services over the Internet. A client program connecting to a web service can read the WSDL to determine what operations are available on the server. Any special data types used are embedded in the WSDL file in the form of XML Schema. Fig. 2 summarizes the basic components of the WSDL file where:

- **Service:** The service can be thought of as a container for a set of system functions that have been exposed to the web based protocols.
- **Port/Endpoint:** The port does nothing rather than defining the address or connection point to a web service. It is typically represented by a simple http URL string. It has been renamed to <endpoint> in WSDL 2.0.
- **Binding:** Specifies the interface, defines the SOAP binding style(RPC/Document) and transport protocol(SOAP).The binding section also defines the operations.
- **Port Type/Interface:** The <port Type> element, which has been renamed to <interface> in WSDL 2.0, defines a web service, the operations that can be performed, and the messages that are used to perform the operation.
- **Operation:** Each operation can be compared to a method or function call in a traditional programming language.
- **Types:** The purpose of the types in WSDL is to describe the data. XML Schema is used (inline or referenced) for this purpose.

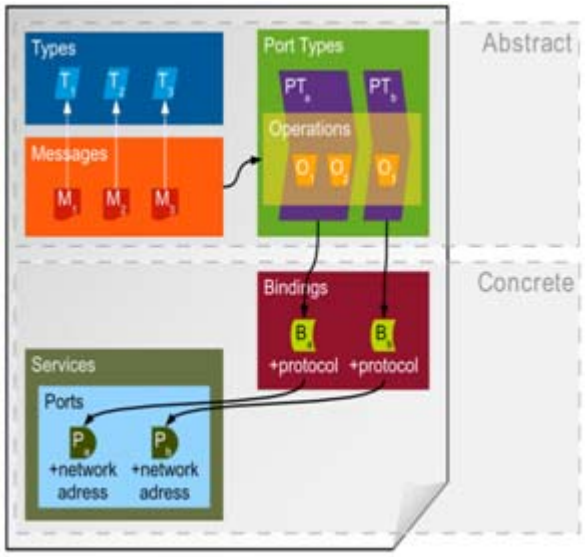

Figure 2: WSDL Basic Components

3. Web Ontology Language for Services

The OWL-S authors target to enable automatic web services discovery, invocation, composition, and interoperation. Fig. 3 shows the basic components of the service class used to describe the web services using OWL-S. There are three main components:

1. The service profiles that describe the function of the service and provide the all necessary information that helps in the discovery process and answer the question "What does this service do?"
2. The service model that describes all the processes the service is composed of, how these processes are executed, under which conditions they are executed and answer the question "How does this service work?"
3. The service grounding that plays the role of coordinator of the service usage. Therefore, it is responsible for the protocols and mapping with traditional web service standards such as WSDL and SOAP.

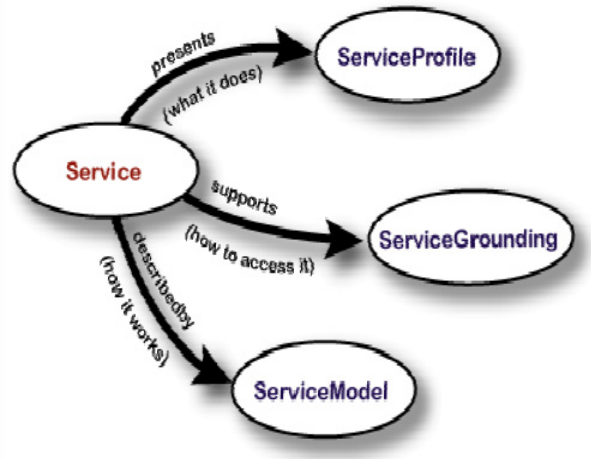

Figure 3: Top level of the service ontology

4. Ontology Search and Standardization Engine

This engine takes in the consideration of semantic web standardization. Fig. 4 presents the OSSE. The input is a required concept with certain features (properties) and the output is an ordered list of possible related ontologies. The concept request is formalized in form of a temporary OWL ontology that contains one concept. The name and the structure of the required concept is the primary supplied information used by OSSE. The engine has three main stages: linguistic search, structural refining, and statistical refining. These three stages are described below:

1. Linguistic Search: In this type of search, OSSE uses the text mining techniques to extract the most important keywords in the concept request. For example, if the system needs to find "car" concept, it should search also for many synonyms and related words (e.g., auto, automobile, machine, motor- car, four-wheel drive, 4WD, and railcar). The list of ontologies is arranged based on the keywords that they contain in terms of term frequency. For each ontology, the summation of term frequency values of each keyword that belongs to the concept request keywords list and belongs to the ontology at the same time is computed. This summation represents a measure of the degree of the ontology linguistic relevance (OLR). Degree of OLR can be calculated by;

$$OLR = \sum_{i=0}^{i=NCK} \text{get TF}(k_i)$$

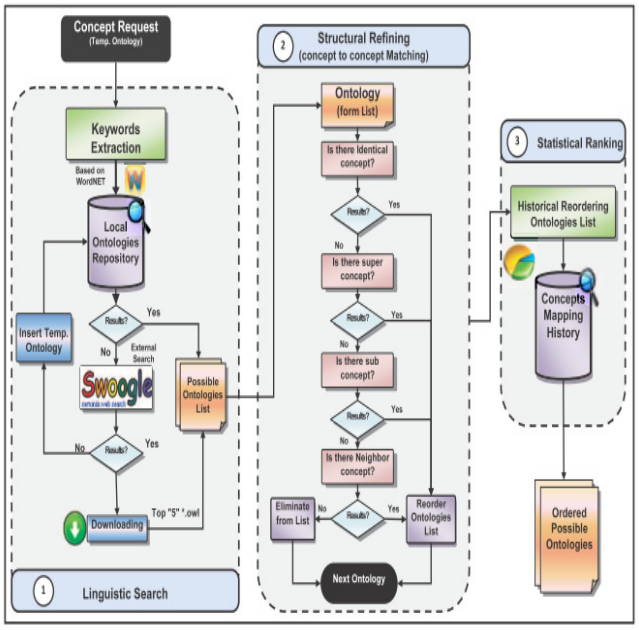

Figure 4: Architecture of OSSE

2. Structural refining. In this stage, OSSE refines the list produced by the linguistic search. This refining is per- formed by searching in each ontology in the list to find any concept related to the required concept. If OSSE does not find any related concept in a particular ontology, this ontology is deleted from the possible ontologies list. "Data concerning the logical structure" which are collected using the inserting methodology, are considered to be the base of the structural refining. We present a simple example to clarify what "Related ontology" means; a simple example is designed to clear up that expression. We assume that the requested concept is called "C_R" and it has three properties {P1, P2, and P3}. "C_R" is compared against another concept called "C" which is part of four suggested concepts-trees that represent all the possible concepts structure in any ontology.

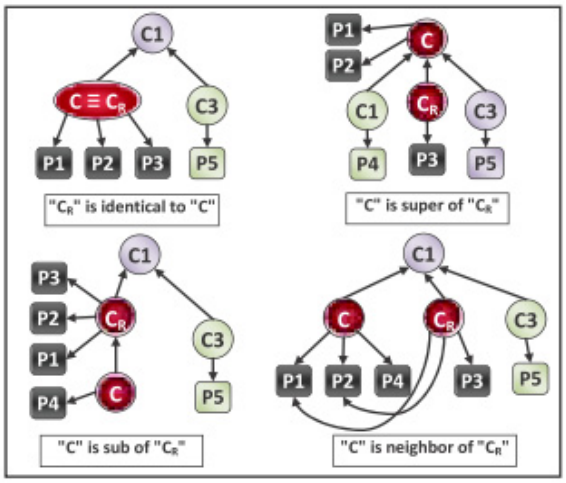

Figure 5: Ontology Concepts trees

1) **Identical Relation:** "C_R " and "C" have the same properties;
2) **Super Relation:** "C" may be considered as a parent of "C_R ";
3) **Sub Relation:** "C" may be considered as a child of "C_R ";
4) **Neighbor Relation:** "C" and "C_R" have some common properties.

3. Statistical Refining: The choices of the service provider are stored in the "Concepts Mapping History" database. These data are used by OSSE to re rank the possible ontologies list. If there are two ontologies with the same rank in the previous step, OSSE uses this historical data to know the most preferred ontologies for the services providers.

4.1 Local Ontology Repository

Our repository is built based on the object-oriented paradigm (OOP) which can be considered as one of the most remarkable approaches to describe the ontologies relationships to adapt existing object-oriented software development methodologies to the task of ontology development, selected approaches are originated from research in artificial intelligence, knowledge representation, and object modeling are presented. In [4], an approach to object-ontology mapping is presented and JAVA is used for implementation and validation. As in OOP, OWL ontology classes and properties are defined as instances of appropriate built-in classes (e.g., owl: Class or owl: Object Property). So, the model of OWL ontologies is very similar to OOP.

4.2 Inserting New Ontology Methodology

The main target of inserting a new ontology to our repository process is to collect two types of data: 1) Data concerning the logical structure, 2) Keywords relevant to the new ontology. These data are used by the OSSE to respond to the requests for the concepts. The first type of data are used by the "structural refining" stage of OSSE; while the second type is used by the "linguistic search" stage of OSSE. As shown in Fig. 6, two concurrent processes collect these two types of data: the first process is called "Structure Extraction." During this process, data concerning ontology concepts (classes), properties, and relationships are collected. The second process is called "Keywords Extraction" where text-mining techniques will be used to extract the keywords form the whole OWL file. In this process, we deal with the OWL file as a text file that contains information about the inserted SWS. There is no doubt that processing the whole file may be time consuming but it provides a clearer vision than processing specific portions of it. This process consists of five steps:

1. Tokenization: is the process of splitting the text into very simple tokens such as numbers, punctuation, and words of different types. In this context, we concentrate on word tokens only.
2. Lemmatization: is the process of reducing inflectional forms and sometimes derivationally related forms of a word to a common base form. For instance: am, are, is) be, cars, car's, cars') car, best, better) good. We use WordNet [5] to perform this process.
3. Now, we have a long list of words. So, we remove the extremely common words which are called "stop words" such as (a, an, am, will, he, she, their; . . . , etc). We depend on Google 5 list 6 which contains 659 words.
4. Then, we remove any word that appears in the keywords list of the languages used to write the ontology such as (XSD [11], OWL [19], RDF [26], RDFS [27]; . . . , etc).
5. Finally, we compute the term frequency (TF) for each word in the list. To demonstrate how to calculate TF, consider a document containing 100 words wherein the word cow appears three times. Then, the TF for cow is then ð3=100Þ ¼ 0:03.

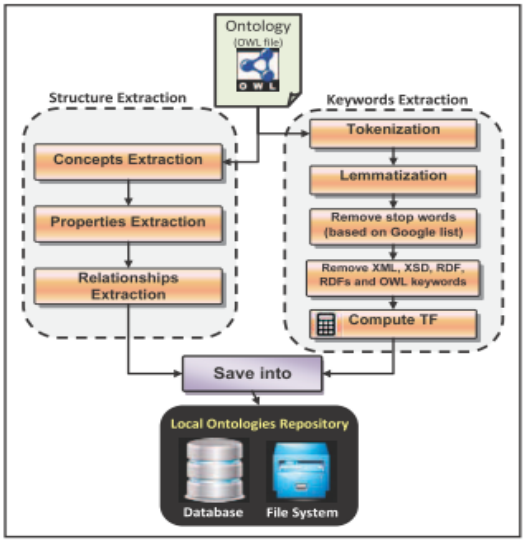

Figure 6: Inserting New Ontology to "Local Ontology Repository"

The ontology contents are stored in the repository file system that can be considered as the original source of information for any ontology. The database is a mean to speed up the process of finding the required ontology information for the SWSs discovery process. The Semantic Mapping of Concepts can be achieved through mapping the WSDL files into their corresponding OWL-S files. The OWL-S supports more automation on complete generation of Information.

5. Semantic Mapping Algorithm

Almost all existing web services are described using WSDL, which is a non semantic definition language. To benefit from the advantages of Semantic Web systems, the service provider needs to redefine his service using semantic annotations. This redefinition can occur by mapping from WDSL to a semantic definition language (e.g., OWL-s). The problems of wasted time, non accurate mapping and absence of any standardization may be the most important issues that the service provider concerns about. Our proposed algorithm helps the service provider to redefine his service using OWL-S. No one can propose a fully automatic algorithm because WSDL, by nature, lacks many information that is needed to have a complete semantic definition. Therefore, the proposed algorithm can be considered as a semi automated tool to help the service provider. During the mapping process, WSDL types are rewritten using OWL and standardized using the OSSE component.

5.1 Mapping Abstract View

Fig. 7 represents the roadmap of our proposed algorithm; it depicts the main components of WSDL and OWL-S files and shows the relations between them. The figure also shows which information can be extracted automatically by the system, which should be supplied by the service provider and which is out of our scope. As stated before, OWL-S concentrates on the description of functional properties of SWS. But, it cannot be used to describe the QoS and other nonfunctional properties. In [3], OWL-Q is presented as an extension to OWL-S which will help the service provider to describe the nonfunctional properties of his service. So, we add a special section in the service profile for any additional QoS parameters supplied by the service provider.

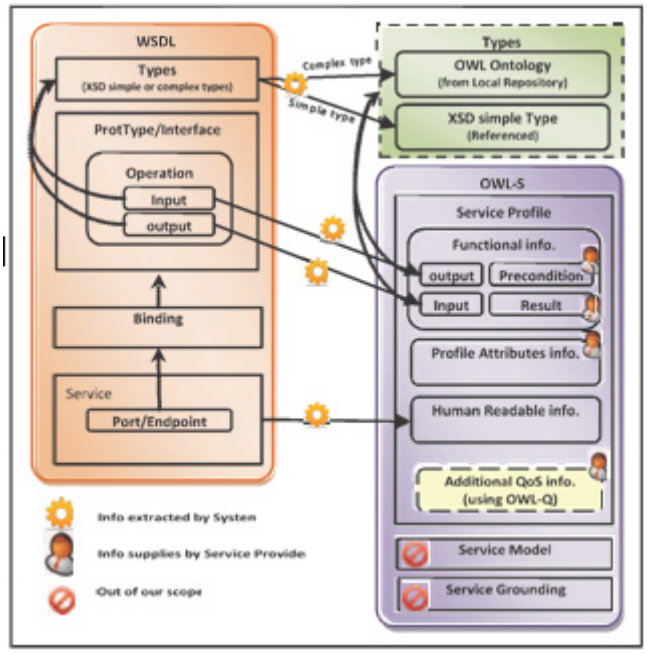

Figure 7: Mapping Abstract View

5.2 The Proposed Semantic Mapping Algorithm

Fig. 8 depicts the proposed semantic mapping algorithm. The algorithm has two phases, the first one is called the "automated phase" in which the WSDL file is parsed to extract information to fill the required properties to construct a service profile in the format of OWL-S. As WSDL does not contain all the required data especially the nonfunctional one, the second phase, called the "manual phase," is designed in a way that the service provider can interpolate the required data.

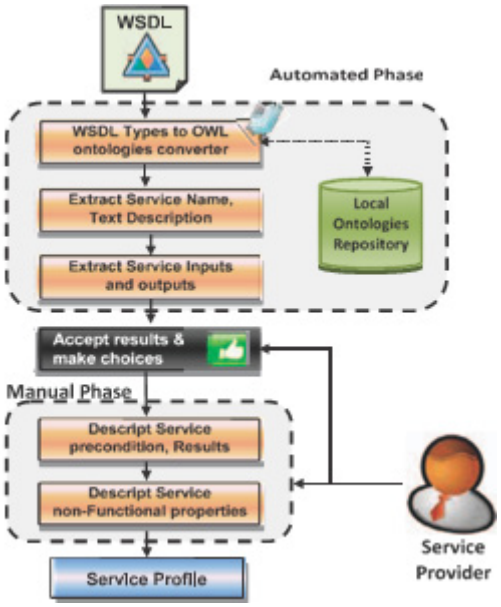

Figure 8: Proposed Semantic Mapping Algorithm

The automated phase consists of three steps:

1. The system recommends ontology for each XSD type in the WSDL file. This process is based on our "Local Ontologies Repository."
2. The system extracts the service name and its text description provided by the service provider.
3. The system extracts the inputs and the outputs of the service.

In manual phase, the service provider will be asked by the system to:

1. Define the preconditions and the effects of the service if found. Define the nonfunctional properties of the services such as (cost, response time, location; …, etc).
2. A tool that applies this mapping algorithm is implemented. This tool accepts the WSDL file then it applies the automated phase of the mapping process then the results of this phase are presented to the service provider. Using the GUI interface of this tool, the service provider reviews the results and chooses the most appropriate ontology concept for each data type. The list of the possible corresponding concepts for each data type is prepared by the "Types Converter" component.

6. Type Convertor Algorithm

One of the most important steps in the process of mapping WSDL to OWL-S is the conversion from the WSDL types, typically XSD types, to OWL ontologies. There are two categories of XSD types: 1) Primitive (simple) XSD types, e.g., string, integer that need to be converted to OWL ontologies. They are defined directly as inputs or outputs of an atomic process in the service model file; 2) Complex XSD types which are translated into OWL ontology concepts whose properties correspond to the elements in the translated type.

According to, the converter of complex XSD types has two alternative designs. The first one is to generate OWL-S specifications that make use of XSD types and make no use of OWL ontologies. The second alternative is to generate concepts that could be totally unrelated to ontologies available in the Semantic Web and therefore they are definitely useless from the automatic reasoning point of view. From our point of view, there is a third alternative that is to search into the Local Ontology Repository using the OSSE component to find a ranked list of the most related ontologies that already exist. Then, ask the service provider to choose the most suitable ontology concept. If the service provider does not accept any of the already existing ontologies, the system uses the second alternative.

Fig. 9 presents the steps of the conversion process. The XSD types are extracted one by one from the WSDL file. The first question is "Is this primitive or complex type?" If it is primitive the conversion process does not work and keeps it as it is. On the other hand, if it is a complex type the converter starts a process to find the most suitable OWL ontology. The process starts by creating a temporary ontology where each XSD complex type can be described by ontology has one concept and many properties.

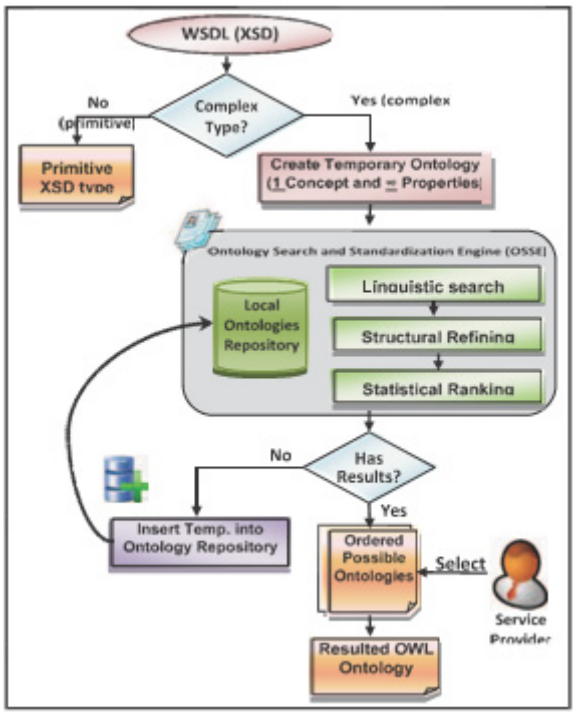

Figure 9: Type Convertor Algorithm

This concept represents the input of the OSSE component. The output of OSSE is a ranked related ontologies list. This list is offered to the service provider to choose the most suitable ontology. In some rare cases, OSSE fails to find any related ontologies. In this case, the system inserts the temporary ontology to the local ontologies repository using the insertion methodology.

7. Results and Mapping Examples

The proposed semantic mapping algorithm is implemented. For testing purposes, we use many WSDL files as a case study to monitor the mapping process (specially the automatic phase) and to evaluate its results. Some of these files belong to OWLS-TC which provides the WSDL file and its corresponding OWLS file. This collection is used to compare the results of our mapping process with those already included within the collection. The rest of the WSDL files belong to real web services (e.g., Yahoo! Mail web service) which help us to study the behavior of our algorithm in a practical environment. Next, we discuss the results of the mapping process and present one example from each group of WSDL files.

7.1 OWLS-TC

Validating the proposed algorithm is done by using 1,000 of WSDL files which are included in OWLS-TC v3.0 and mapped using the proposed algorithm. The automatic phase of the mapping process is done within 1 hour. This time can be neglected when compared with the time consumed in the case of manual mapping.

Fig. 10 shows a comparison between the WSDL to OWL-S tool and the proposed mapping algorithm. The figure presents the relation between the number of concepts generated by the system and the number of registered services. We can obviously note that the number of used concepts will be increased when the number of services increases in both cases.

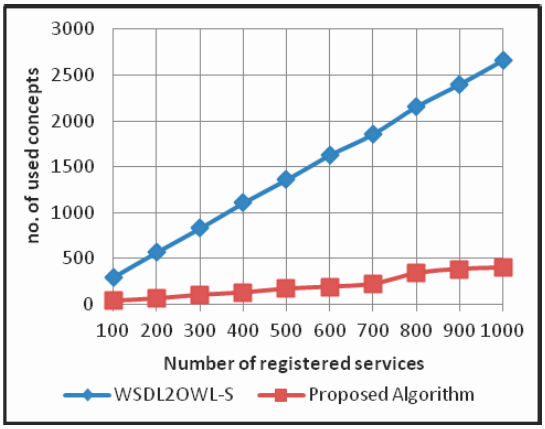

Figure 10: No of used concepts versus no. of registered concepts

But, in the case of WSDL2OWL-S tool the number of concepts increase with very high rate when compared to the case of our proposed algorithm. For example, when the number of registered services becomes 1,000, the average number of concepts per service is 2.66 in case of WSDL2OWL-S and 0.4 in case of the proposed algorithm. So, the proposed algorithm is more scalable than WSDL2OWL-S. It is important to state that the performance of the discovery process is negatively affected when the number of concepts defined in the system increases. That is because the study of inputs/outputs matching between the request and the available services is a major task for any discovery process. There is no doubt that this matching become faster and more accurate when the total number of concepts which defines the inputs and outputs type becomes smaller. So, the proposed semantic mapping algorithm will be better than WSDL2OWL-S from the discovery point of view.

8. Conclusion and Future Directions

In this study, we target to solve the problem of enabling web services discovery. We propose a semantic mapping algorithm that helps to facilitate the integration of the current conventional web services into the new environment of the Semantic Web. This has been achieved by extracting information from WSDL files and using it to create a new semantic description files using OWL-S.

The proposed algorithm contains a basic component called “Types Converter” which is used to convert XSD complex types to ontologies. This converter depends on OSSE component that uses the “Local Ontology Repository” to find a suitable ontology for each XSD complex type. The bottom-up design approach is used to implement and validate our mapping algorithm. So, we start with creating the “local ontology repository” then we implement the OSSE component and finally we implement the type converter and the mapping algorithm. Many designed and real examples are used to verify the applicability of the proposed algorithm. The results prove that this algorithm helps the service providers to cut down the effort and the time consumed

in redefining their services in a semantic manner.

The proposed semantic mapping algorithm represents a starting point to transform our proposed intelligent discovery mechanism into an applicable one. Without trying to benefit from the already available web services, any semantic discovery mechanism will face many problems to be widely used. So, the main task for this algorithm is to help the service providers to re describe their WSs in a semantic manner.

Experimental results show that the proposed algorithm has the potential to significantly reduce the time and efforts of the mapping process. Moreover, they show that the algorithm consideration of the standardization problem promises that it will have a positive impact on the discovery process as a whole. In the future, we will continue to implement and validate the rest of our proposed discovery mechanism components.

References

[1] M. Burstein, C. Bussler, M.Zaremba, T.Finin, M.N. Huhns, M.Paolucci, A.P.Shet and S illiams, "A Semantic Web Services Architecture," IEEE Internet Computing, vol. 9, no. 5, pp.
[2] T. Berners-Lee, J. Hendler, and O. Lassila, "The SemanticWeb,"Scientific Am. Magazine, vol. 284, no. 5, pp. 34-43, 2001.
[3] J. Cardoso, Semantic Web Services: Theory, Tools, and Applications. Idea Group, Inc., 2007.
[4] B. Sapkota, D. Roman, and D. Fensel, "Distributed Web Service Discovery Architecture," Proc. Advanced Int'l Conf. Telecomm. and Int'l Conf. Internet and Web Applications and Services (AICT- ICIW '06),2006
[5] T.A. Farrag and H.A. Ali, "A Cluster-Based Semantic Web Services Discovery and Classification," Proc. ACME Second Int'l Conf. Advanced Computer Theory and Eng., pp. 1825- 1834, 2009.
[6] Web Services Description Language (WSDL), W3C in web
[7] Web Ontology Language for Services (OWL-S), http://www.w3.org/Submission/OWL-S/, 2004.
[8] D. Martin, M. Burstein, D. Mcdermott, S. Mcilraith, M. Paolucci, K.Sycara, D.L. Mcguinness, E. Sirin, and N. Srinivasan, "Bringing Semantics to Web Services with OWL-S" World Wide Web, vol. 10, no. 3, pp. 243-277, Sept. 2007.
[9] Extensible Markup Language (XML), W3C Recommendation: http://www.w3.org/XML/, 2012.
[10] Simple Object Access Protocol (SOAP) Version 1.2, W3C Recom- mendation (second ed.) http://www.w3.org/TR/soap/, Apr.2007.

Author Profile

Rajalakshmi. A received B. Tech degree in Information technology from Anna University 2011..currently doing M.E degree in Computer Science and Engineering and project work related to web services.

Priyaradhikadevi. T received M.Tech degree in Computer Science from Anna University. Pursuing Ph.D and published many international and national journals. Two books has been published. Participated in many national and international conferences.